Pasi Hulkko

Verkkosivustojen responsiivinen suunnittelu ja kehitys

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Mediatekniikan koulutusohjelma

Insinöörityö

16.10.2014

Metropolia

Tekijä Otsikko Sivumäärä Aika	Pasi Hulkko Verkkosivustojen responsiivinen suunnittelu ja kehitys 36 sivua 16.10.2014
Tutkinto	insinööri (AMK)
Koulutusohjelma	mediatekniikka
Suuntautumisvaihtoehto	digitaalinen media
Ohjaajat	Projektipäällikkö Christer Lybeck Yliopettaja Harri Airaksinen

Insinöörityön tavoitteena on tutkia responsiivisten verkkosivujen toteutusta ja siihen liittyviä menetelmiä ja työtapoja. Tilaajayritykselle toteutettiin kolme responsiivista verkkosivuprojektia, joiden aikana yrityksen työtapoja responsiivisten verkkosivujen osalta kehitettiin.

Responsiiviset toteutukset tarkoittavat verkkosivujen toteutusta, joka ottaa huomioon kaikki laitteet pienistä älypuhelimista isoihin työpöytätietokoneisiin. Responsiiviset verkkosivut ovat yleistyneet nopeasti pieninäyttöisten laitteiden kasvun myötä. Responsiiviset sivut mukautuvat päätelaitteen näytön kokoon sopivaksi, mikä luo miellyttävän käyttökokemuksen kaikille käyttäjille.

Insinöörityö sisälsi responsiivisten verkkosivujen toteutuksen suunnittelusta tekniseen toteutukseen. Teknisen toteutuksen mahdollistavat modernit verkkoteknologiat, kuten HTML5, CSS3 ja JavaScript. Projekteissa käytettiin responsiivisiin toteutuksiin sopivia menetelmiä, kuten SCRUM.

Insinöörityön toteutuksen pohjana olleiden kolmen responsiivisen työn tuloksena responsiivisten verkkosivujen toteutukseen liittyvistä työtavoista ja tekniikoista opittiin paljon tulevia projekteja silmälläpitäen. Projektien toteutuksien aikana huomattiin, että sellaisista työkaluista kuin Twitter Bootstrap ja SASS on huomattavaa hyötyä nopeatempoisessa responsiivisessa kehityksessä. Myös projektien huolellinen suunnittelu on ensiarvoisen tärkeää responsiivisissa projekteissa.

Avainsanat	responsiivinen kehitys, responsiivinen suunnittelu, SASS, Bootstrap, SCRUM, mobiilikehitys

Author	Pasi Hulkko
Title	Design and implementation of responsive websites
Number of Pages	36 pages
Date	16 October 2014
Degree	Bachelor of Engineering
Degree Programme	Media Technology
Specialisation option	Digital Media
Instructors	Christer Lybeck, Project Manager Harri Airaksinen, Principal Lecturer

The purpose of this thesis is to study the design and implementation of responsive websites. The study consisted of three different websites, which were designed and implemented for the employer company.

Responsive websites attempt to take into consideration all devices from small mobile devices to larger desktop computers. Responsive websites have grown in popularity following the growing number of small touchscreen devices. Responsive websites adapt to fit the needs of the device, offering a comfortable user experience for all users.

The thesis included technical implementation and design of responsive websites. The technical implementation was carried out using modern web technologies such as HTML5, CSS3 and JavaScript. Project management strategies suitable for agile development such as SCRUM, were used.

The three projects, which were implemented in this thesis helped improve the workflow and techniques for future projects. During the projects, it became clear that tools and techniques such as Twitter Bootstrap and SASS, greatly improved the fast paced development of responsive websites. In addition, thorough planning and designing is critical in responsive projects.

Keywords	Responsive design, SASS, Bootstrap, SCRUM, Mobile development

Sisällys

Lyhenteet

CSS	Cascading style sheets. Verkkodokumenttien tyylittelyyn kehitetty kieli.
SASS	Syntactically Awesome Style Sheets. Tyylittelykieli, joka tulkataan CSS:ksi Rubyn avulla.
HTML	Hypertext Markup Language. Verkkodokumenttien merkintäkieli.
TFS	Team Foundation Server. Microsoftin versionhallintajärjestelmä.
WAP	Wireless Application Protocol. Varhainen protokolla langattomille sovelluksille

1 Johdanto

Tämän insinöörityön tavoitteena on tutkia responsiivisten internetsivustojen toteutusta suunnittelun ja teknisten välineiden näkökulmasta. Työssä tutkitaan responsiivisen toteutuksen eri vaiheita ja pyritään avaamaan ne aina suunnittelusta, toteutukseen ja lopuksi testaamiseen.

Työssä tutkitaan responsiivisuutta kolmen eri projektin kautta. Kaikissa projekteissa oli erilaiset lähtökohdat ja haasteet, ja niissä käytettiin hieman erilaisia työkaluja ja tekniikoita. Kaikki projektit toteutetaan Episerver-sisällönhallintajärjestelmään, mutta insinöörityö keskittyy vain toteutuksien käyttöliittymiin.

Insinöörityö on toteutettu Creuna Finlandille tavallisten asiakasprojektien ohella. Creuna Finland on osa pohjoismaista Creuna-konsernia, joka on pohjoismaiden suurin digitoimisto. Creunan palveluksessa on yli 370 työntekijää Norjassa, Tanskassa, Ruotsissa ja Suomessa. Creunan vahvuutena on vahva pohjoismainen design-osaaminen yhdistettynä vahvaan teknologiaosaamiseen. Yrityksen tavoitteena on tarjota digitaalisen median osaamista kaikilta sen osa-alueilta.

Insinöörityön alussa esitellään responsiivisen kehityksen historiaa. Projektit esitellään ennen teoriaosuutta ja niiden toteutukseen palataan myöhemmin. Projektien esittelyn jälkeen käsitellään responsiivisten toteutuksien suunnittelua, niihin käytettäviä tekniikoita ja testausta.

Insinöörityöraportissa responsiivisuuden taustaa ja merkitystä pyritään havainnollistamaan kuvilla, jotka kuvaavat erilaisten päätelaitteiden ja varsinkin mobiililaitteiden monimuotoisuutta ja suurta kasvua. Responsiivisen kehityksen erilaiset työtavat pyritään selittämään havainnollisesti ja responsiivisen toteutuksen eri vaiheita pyritään käsittelemään niiden haasteiden ja mahdollisten ratkaisujen kautta.

2 Responsiivisen suunnittelun historia

Verkkosivujen responsiivinen suunnittelu ja kehitys on ollut suuressa kasvussa jo muutaman vuoden ajan. Erikokoisten laitteiden yleistyminen ja niiden verkkoselaimien kehittyminen on johtanut siihen, että verkkosivujen ylläpitäjien on entistä vaikeampaa ennustaa käyttäjien päätelaitteiden resoluutiota, käyttöjärjestelmää tai selainta. Perinteisesti mobiili- ja työpöytä-laitteet ovat olleet käyttöliittymäkehityksen kannalta toistensa ääripäitä, joihin internetsivujen optimoinnissa on tähdätty kahdella versiolla, mobiiliversiolla ja työpöytäversiolla. Nykyään näiden kahden ääripään version välille mahtuu paljon eri laitteita ja selaimia, joilla sisällön voisi näyttää paremminkin. [1, s. 1.]

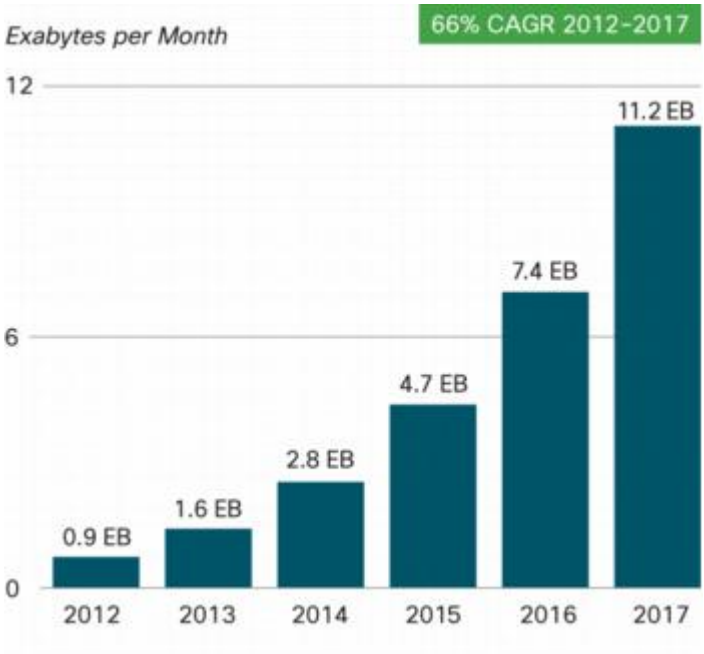
Mobiililaitteiden internetin selailumahdollisuudet olivat alun perin hyvin rajalliset. 2000-luvun alussa mobiililaitteet olivat vielä kaukana siitä, mitä ne ovat nykyään. Kosketusnäyttöisiä puhelimia ei ollut vielä markkinoilla, ja näyttöjen koot olivat todella pienet verrattuna nykyään saatavilla oleviin puhelimiin. WAP-tekniikka oli senaikaisten mobiililaitteiden tapa päästä internetiin, ja monista isoista palveluista olikin WAP-versio olemassa. WAP-tekniikka oli kuitenkin hidas, ja sen tarjoamat mahdollisuudet rajalliset. Se antoi verkkosivuille lähinnä mahdollisuuden esittää sisällön XML-muodossa. [2.]

Muutama vuosi WAPista eteenpäin markkinoille alkoi ilmestyä laitteita, joissa oli kehittyneemmät verkkoselaimet. Nämä laitteet perustuivat esimerkiksi Nokian Symbian-käyttöjärjestelmään. Ensimmäisten mobiiliselainten ongelmana oli niiden rajallinen HTML-, CSS- ja JavaScript-tuki. Osittain mobiiliselainten vaihtelevien standarditukien ansiosta verkkopalveluista ruvettiin tekemään erillisiä mobiiliversioita. Verkkosivujen ulkoasuista saatettiin tehdä kaksi eri versiota: täysi versio työpöytälaitteille ja riisuttu versio mobiililaitteille. Näin ollen verkkosivujen koodista jouduttiin ylläpitämään kahta eri versiota. Esimerkiksi MTV3:n verkkosivujen mobiiliversio on osoitteessa m.mtv3.fi, kun tavalliselle sivulle pääsee mtv3.fi-osoitteella. Varhaisten kosketusnäyttöisten puhelimien ongelmana oli myös heikko käytettävyys, joka haittasi niiden hyväksymistä varteenotettaviksi laitteiksi internetin selailuun suuren yleisön keskuudessa. [2.]

Responsiivisuuden mahdollistavaa media kysely -tekniikkaa oli suunniteltu jo varhain vuonna 2002, mutta se tuli virallisesti käyttöön vasta vuonna 2012 CSS3-standardiin liittyvän moduulin myötä. Media kyselyt mahdollistivat ulkoasun mukautumisen näytön koon mukaan. Mobiililaitteille voitiin nyt tietyissä näytön ko'oissa jakaa erilainen

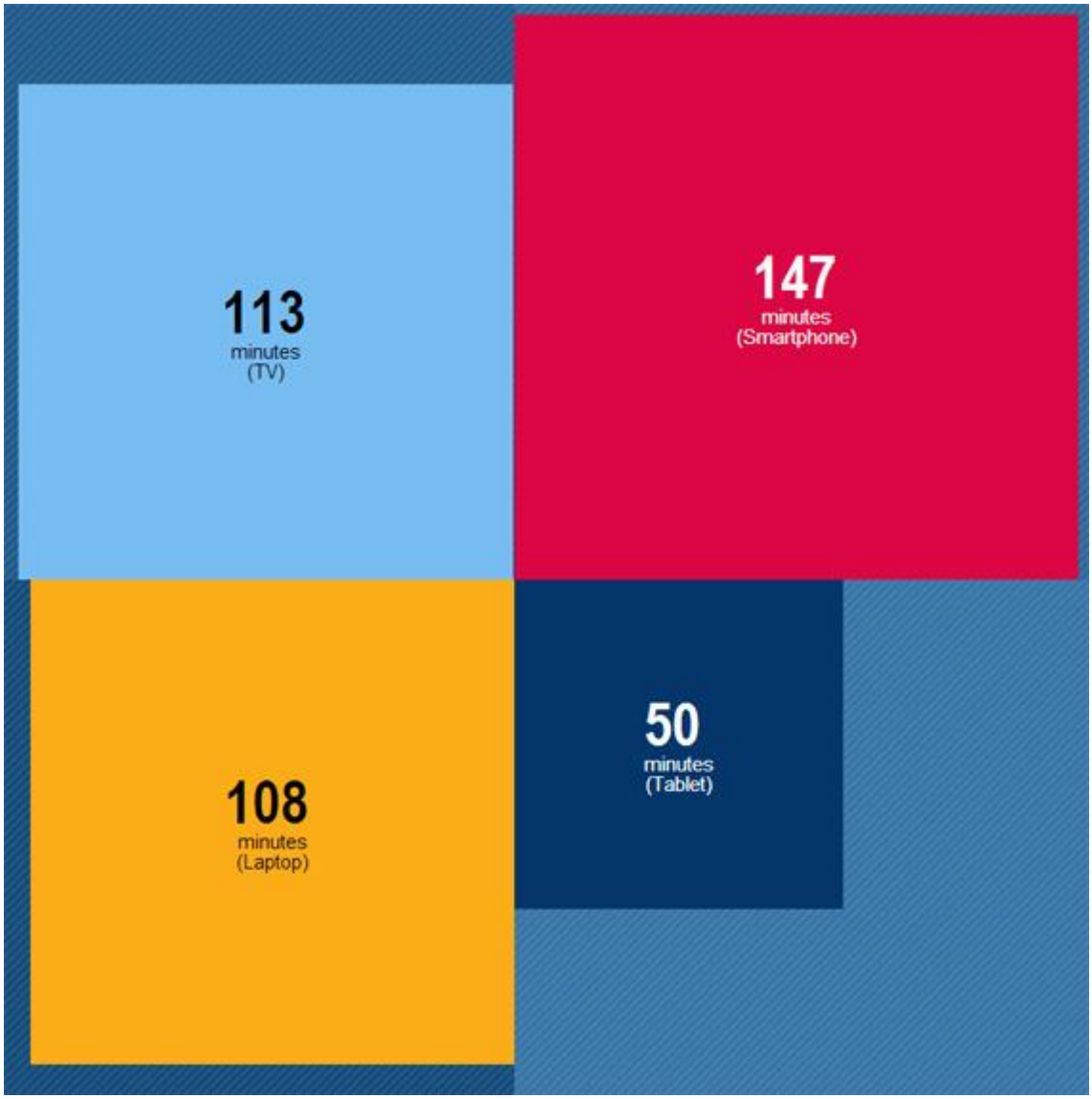
ulkoasu ja toisin kuin mobiili vastaan työpöytä -asetelmassa, myös niiden väliin jääville laitteille voitiin tarjota optimoitu käyttäjäkokemus. [3.]

Vuoden 2007 aikana Apple julkisti ensimmäisen version iPhone – kosketusnäyttöpuhelimesta, joka oli aikanaan reilusti edellä esimerkiksi Nokian älypuhelimia. iPhonen helppokäyttöisyyden myötä käyttäjät ja kehittäjät ymmärsivät, että verkkopalveluja voi käyttää sujuvasti myös mobiililaitteella. Vuodesta 2007 eteenpäin mobiililaitteiden määrä verkossa on kasvanut paljon ja responsiivisen suunnittelun aikajana voidaankin jakaa kahteen osaan: aikaan ennen iPhonea ja aikaan iPhonen jälkeen. Myös nopeiden verkkoyhteyksien, kuten 3G ja 4G, kehitys on kasvattanut mobiililaitteiden määrää verkossa. Datan määrä mobiiliverkoissa on kasvanut paljon nopeiden yhteyksien yleistyessä. Verrattuna esimerkiksi vuoteen 2012, kuvan 1 mukaisesti, Cisco ennustaa mobiilidatan määrän kasvavan yli kymmenkertaiseksi vuoteen 2017 mennessä. [1, s. 10; 4.]



Kuva 1. Ciscon ennuste mobiilidatan kuukausittaisesta määrästä vuoteen 2017 asti eksatavuissa.[4]

MillwardBrownin vuonna 2014 julkaiseman raportin mukaan mobiilinäyttöjen käyttöaika on ohittanut kaikki muut näytöt, kuten television ja kannettavat tietokoneet. Kuvassa 2 näkyvät eri näyttöjen saamat osuudet ihmisten huomiosta. [5.]



Kuva 2. Kuluttajien eri päätelaitteiden kanssa viettämä aika minuutteina päivässä per henkilö[5].

3 Projektien esittely

Insinöörityön projekteina oli kolme Creuna Finlandille toteutettua responsiivista sivua tai sivustoa. Projektien toteutukset ajoittuivat vuoden 2013 lopulle. Kaksi projekteista oli Creuna Finlandin asiakkaan yritys A:n kampanjasivustoja ja samalla Creuna Finlandin ensimmäisiä responsiivisia toteutuksia. Yksi projekti oli responsiivinen ”kasvojen kohotus” yritys B:n verkkopalvelulle.

Creuna toteutti yritys A:n sivustouudistuksen vuoden 2013 aikana. yritys A:n vanhat sivut olivat vanhentuneet ja kilpailijoista jäljessä. Lisäksi niiden toteutuksessa ei ollut otettu mobiilikäyttäjiä huomioon. Uusi sivusto toteutettiin Episerver-sisällönhallintajärjestelmään ja tälle alustalle toteutettiin myös kaksi reponsiivista kampanjasivustoa.

3.1 Yritys A:n kampanjasivut

Yritys A:n sivustouudistuksen myötä kaiken yritys A:n toiminnan sen verkkotunnuksen alla tuli olla responsiivista. Kampanjasivustojen tuli olla visuaalisesti näyttäviä isoilla laitteilla ja samalla toimia moitteetta mobiililaitteilla. Molempien kampanjoiden haluttiin kiinnittävän kuluttajien huomion, ja samalla haluttiin aktivoida kuluttajia osallistumaan yritys A:n uusiin tuotteisiin liittyviin kilpailuihin.

Yritys A toi kesällä 2013 markkinoille uuden kuluttajille suunnatun tuotteen. Koska tuote oli uusi, katsottiin, että kuluttajien piti heti sivulle tultuaan ymmärtää, mikä tuotteen idea on. Kampanjasivun haluttiin myös rohkaisevan kuluttajia toteuttamaan omia versioita tuotteesta.

Syksyllä 2013 yritys A julkaisi uuden toisen kuluttajille suunnatun tuotteen. Creuna Finland sai toimeksiannon toteuttaa tuotteen julkaisua varten kampanjasivuston, sekä kolme videota, joita käytettäisiin sivulla. Yritys A:lla oli toisen yrityksen tekemä valmis televisioon tarkoitettu mainos, jota myös haluttiin hyödyntää sivulla. Kampanjasivuston teemana oli helppokäyttöisyys, ja sitä tukemassa oli tuotteen kampanjaan tarkoitettu viesti, jota haluttiin hyödyntää kampanjasivulla.

3.2 Yritys B:n verkkosivusto

Creuna Finland toteutti yritys B:n tuotetta esittelevälle sivustolle päivityksen EPiServer 7 -versioon, verkkokaupan EPiServer Commerce -lisäosan avulla ja responsiivisen kasvojenkohotuksen. Yritys B:n verkkosivusto oli vahvasti brändätty ja sivuston ulkoasu tahdottiin säilyttää mahdollisimman ennallaan. Sivuston responsiivisen kasvojenkohotuksen haluttiin myötäilevän yrityksen jo olemassa olevaa brändiohjetta sen tuotteelle. Verkkosivustolle haluttiin myös verkkokauppa, joka toteutettiin EpiServer Commerce -lisäosan avulla ja jonka piti myös toimia mobiililaitteilla. Creunan rooli tässä projektissa oli lähes kokonaan tekninen, koska yritys B halusi itse määritellä visuaalisen toteutuksen. Projektin hallinnassa päädyttiin käyttämään ketterään ohjelmistokehitykseen tarkoitettua scrum-menetelmää.

4 Responsiivinen internetsivu

Responsiivisella internetsivulla tarkoitetaan sivua, joka pyritään näyttämään mahdollisimman hyvin kaikilla päätelaitteilla. Responsiivisessa suunnittelussa oletuksena on, että käyttäjällä saattaa olla käytössä mikä tahansa laite, jossa on internetselain, koska erikokoisia laitteita on nykyään suuri määrä. [2, s. 8.]

Responsiivisen sivun mahdollistaa joukko tekniikoita, joilla selaimelle saadaan haluttu tyyli ja sisältö. Tekniikoiden lisäksi responsiivinen suunnittelu sisältää joukon hyväksi todettuja työtapoja ja suunnittelumalleja. Lisäksi responsiivisuuteen liittyy paljon tekniikoita ja tapoja, joita ei ole kehitetty pelkästään responsiivista suunnittelua silmälläpitäen.

Verkkosivustoja ohjelmoitaessa ei voida tietää, millaisella päätelaitteella loppukäyttäjä sivua lopulta katselee. Nykyään eri päätelaitteita on paljon ja jokainen esittää verkkosivut eri tavalla – joko laitteen fyysisen koon tai käytettävän selaimen takia. [2, s. 9.]

Verkkosivustojen suunnittelu lainaa paljon asioita perinteisen painomedian ammattilaisilta – onhan suuri osa esimerkiksi graafisista suunnittelijoista alun perin työskennellyt perinteisissä painotaloissa tai painomedian kanssa. Painettaessa paperille tiedetään loppukäyttäjän paperin koko, ja näin ollen voidaan sommittelu tehdä tarkasti ja tarkalleen yksille mitoille. [6, s. 2.]

Yhdelle koolle suunnittelu oli yleistä vielä suurimman osan 2000-luvun alusta, jolloin laitteiden näyttöjen eri kokoja oli vielä suhteellisen pieni määrä. Kun päätelaitteen koko oli tuntematon, suunnittelijat joutuivat tekemään päätöksen, joka rajasi käyttäjiä jollakin tapaa: pienemmän näytön omistavat käyttäjät saattoivat joutua selaamaan liian isolle näytölle suunniteltua sivua, kun taas ison näytön omistavat eivät aina pystyneet hyödyntämään näytön tuomia etuja. Tässä vanhassa suunnittelumallissa verkkosivut suunniteltiin pienimmän yleisen näytön resoluution mukaan, esimerkiksi 800 pikselin leveyteen. [6, s. 3.; 6, s. 10.]

Ennen CSS3:n tuomia responsiivisia ominaisuuksia verkkopalveluista oli usein erillinen mobiiliversio. Erilliset mobiilisivustot olivat riisuttu versio varsinaisesta sivustosta, esimerkiksi ulkoasun osalta. Kahden eri version ongelmaksi muodostuu se, ettei

käyttäjä saa aina omalle laitteelleen sopivaa versiota, vaikka sivulla olisikin jokin mekanismi, jolla selainta tai laitetta yritettäisiin tunnistaa. Esimerkiksi MTV3:n mobiilisivuston tapauksessa mobiilikäyttäjä saattaisi jakaa linkin johonkin kiinnostavaan uutiseen Facebookissa tai muussa sosiaalisessa mediassa, jolloin linkin saaneet pöytäkonekäyttäjät saisivat myös erillisen mobiiliversion eteensä. Responsiivisella toteutuksella tätä ongelmaa ei ole. Vaikka erillisiä mobiilisivuja edelleen jonkin verran on, responsiiviset sivustot ovat muodostumassa pääasialliseksi toteutustavaksi mobiililaitteissa toimiville internetsivuille. [6, s. 6.]

4.1 Responsiivisessa suunnittelussa käytettävät tekniikat

Responsiivisen internetsivun mahdollistaa pääpiirteittäin kolme eri tekniikkaa. HTML eli Hypertext Markup Language asettaa sisällölle rakenteen ja sen roolin dokumentissa. Responsiivisuuden kannalta dokumentin rakenteella on merkitystä. Esimerkiksi joissain navigaatioratkaisuissa on tärkeää pitää mielessä, että samaa elementtiä voi käyttää uudestaan eri tyyleillä ja näin ollen säästää internetsivun tiedostokoossa. Lisäksi HTML:ää kirjoittaessa olisi syytä pitää mielessä myös esimerkiksi ruudunlukuohjelmat ja muut erikoisemmat laitteet; ne kaikki hyötyvät hyvin jäsennellystä HTML-dokumentista. Responsiivisen suunnittelun tarkoituksena on saattaa internetsivut mahdollisimman suurelle yleisölle, joten on luontevaa, että toteutuksessa otetaan huomioon muutkin asiat kuin näytön koko. [7.]

HTML-merkintäkielen lisäksi responsiivisuuteen tarvitaan CSS-tyylit. CSS:n avulla dokumenttiin saadaan haluttu tyyli, kuten esimerkiksi värimaailma ja sommittelu. CSS-tyylit voidaan kirjoittaa suoraan dokumenttiin, mutta yleensä ne kirjoitetaan erilliseen tiedostoon, jossa on CSS-pääte. CSS-tiedostoihin annetaan viittaus HTML-dokumentissa. CSS-tiedostojen viittauksiin voidaan asettaa myös eri medioiden tyypit, kuten esimerkiksi ”screen”, joka tarkoittaa mitä tahansa näyttöä, tai ”print”, jota käytetään, kun dokumentti halutaan tulostaa. Eri medioiden tyypit tulevat vastaan myös tärkeimmässä tekniikassa, joka mahdollistaa responsiivisuuden: mediakyselyissä (*engl. media query*). [8.]

CSS tekniikat responsiivisessa kehityksessä

Media queryt eli mediakyselyt ovat tärkein työkalu responsiivista internetsivua tehtäessä. Niiden avulla päätelaite ja selain ymmärtävät, mitä tyyliä niiden pitää milloinkin käyttää. Esimerkiksi yritys A:n kampanjasivulla yksi mediakysely alkaa esimerkin 1 mukaisesti:

```
@media (max-width:340px){
.header .ohje-row .ohje-container h2 {
font-size: 1.5em;
}
```

Koodiesimerkki 1. Pieniin laitteisiin tähtäävä mediakysely.

Esimerkin 1 mediakyselyssä annetaan selaimelle tieto, että media kyselyn sisällä olevia tyylejä pitää käyttää silloin, kun laitteen maksimileveys on 340 pikseliä, eli käytännössä tämä tarkoittaa kosketusnäytöllisten puhelinten pienimmässä päässä olevia laitteita, joita pidetään pystyasennossa [8].

Mediakysely voi olla myös mutkikkaampi, kuten Ethan Marcotte on esittänyt teoksessa Responsive Web Design, ks. esimerkki 2.

```
@media screen and(min-device-width: 480px) and (orientation: landscape) {
 …
}
```

Koodiesimerkki 2. Vaakatasossa olevalle tabletille tarkoitettu mediakysely [6, s. 78].

Mediakyselyissä toimivat loogiset operaattorit AND ja OR, joista AND merkitään sanalla ”and” ja OR merkitään yksinkertaisesti pilkulla. Lisäksi mediakyselyissä on muita operaattoreita, kuten not, only, ja lausekkeita, kuten ”device-aspect-ratio”. Jo pelkästään näitä tekniikoita käyttämällä saadaan internetsivu näkymään monelle eri laitteelle hyvin. [8.]

Responsiivista sivustoa suunnitelltaessa pitää päättää, mihin leveyteen eri mediakyselyt tahdostaan kohdentaa. Paikkoja, joissa sivulle syötettävä CSS muuttuu, kutsutaan breakpointeiksi. Responsiivisten sivustojen hyvä suunnittelu on tärkeää, jotta

elementit saadaan mukautumaan pienempiin resoluutioihin mentäessä. Hyvällä suunnittelulla voidaan vähentään tarvittavien breakpointien määrää. Breakpointeja on hyvä olla ainakin kolme: yksi isoille näytöille, yksi keskikokoisille ja yksi pienille näytöille, mutta ei ole mitään teknistä estettä käyttää useampaakin breakpointia.

Mittasuhteiden osalta Ethan Marcotte peräänkuuluttaa niin sanottua joustavaa gridiä. Gridillä tarkoitetaan kehystä, jota käytetään graafisessa suunnittelussa apuvälineenä elementtien sijoittamiseen. Yksinkertaisimmillaan joustava grid tarkoittaa mittasuhteissa sitä, että elementin mitta, esimerkiksi Photoshop-tiedostosta mitattuna, suhteutetaan sen ympäröivään kontekstiin. Yritys A:n toisen tuotteen kampanjasivulla pakettien alkuperäinen konteksti oli koko sivun leveys, josta oli poistettu marginaalit, eli 940 pikseliä. Yhden paketin koko taas oli 300 pikseliä, jolloin yhden paketin leveydeksi saatiin laskutoimituksesta 300 / 900 * 100. Näin pakettielementin kooksi tuli lopulta noin 31.91%. Samaa logiikkaa voi soveltaa fonttien kokoon: kun em-yksikkö on sidottu 16 pikseliin, kontekstina toimii 16, joten esimerkiksi 24 pikselin fontti saadaan em-muotoon laskutoimituksella 24 / 16 * 100. [6, s. 20.]

Responsiivisen gridin voi kirjoittaa itse, mutta kehitystä voi joissakin tapauksissa nopeuttaa käyttämällä jotain valmista, responsiivista CSS-kehystä. Erilaisia CSS-kehyksiä, joilla responsiivisen gridin voi toteuttaa, on monia. Niiden hyötynä on nopea kehitys valmiiden CSS-luokkien avulla, ja varsinkin vakiintuneissa kehyksissä, kuten Twitter Bootstrap (http://getbootstrap.com/) ja esimerkiksi Zurb Foundation (http://foundation.zurb.com/), yhteensopivuus eri selaimien ja laitteiden kanssa on lähes taattu. Ongelmaksi kehyksien kanssa muodostuvat ylimääräiset CSS-luokat, joita joutuu käyttämään HTML-elementeissä. Ylimääräisten luokkien tuoma lisäkoko ei ole nopeuden tai minkään muun teknisen mittarin kannalta merkittävä, mutta kun pyrkimyksenä on tuottaa luettavaa ja selkeää koodia, ylimääräiset luokat ovat häiritseviä. Esimerkiksi Bootstrapia käytettäessä voisi olla seuraavanlainen div-elementti:

```
<div class="span4 offset1 tuote-kuva-container">
```

Elementin luokat kertovat, että sen leveys on 4 osaa gridistä, sitä on vedetty 1 gridin osa oikealle ja lisäksi elementti on tarvinut yksilöidyn luokan tyylittelyä varten [9].

Vaihtoehtona tavallisen CSS:n käyttämiselle voidaan käyttää SASS-kieltä (http://sass-lang.com/). SASS-lyhenne tulee sanoista ”Syntactically Awesome Stylesheets” [10].

SASS on komentosarjakieli, joka käännetään tavalliseksi CSS:ksi. SASS ei sulje pois CSS:n käyttöä, sillä se hyväksyy tavallisen CSS:n syntaksin sellaisenaan, mutta SASS tuo kehittäjän ulottuville hyödyllisiä ominaisuuksia kuten

- muuttujat
- sekoitukset (*engl. mixins*)
- sisentämisen (*engl. nesting*)
- erilaiset funktiot esimerkiksi värien sekoitukseen.

Erityisesti sisentäminen on kehittäjän työtä helpottava ominaisuus. Esimerkiksi kun CSS:llä valittaan div-elementin sisällä olevan span-elementin sisältä a-elementti ja p-elementti erikseen, se voidaan kirjoittaa tavallisesti esimerkin 3 mukaisesti.

```
div span a{
   color: blue;
}
div span p{
    color: black;
}
```

Koodiesimerkki 3. Tavallinen CSS-tyyli p- ja a -elementeille, jotka ovat div- ja span -elementtien sisällä.

SASSia käyttämällä sama lopputulos saadaan koodiesimerkiin 4 mukaisesti:

```
div {
    span{
        p{
            color: black;
        }
        a{
            color: blue;
        }
    }
}
```

Koodiesimerkki 4. SASS-tyyli p- ja a -elementeille, jotka ovat div- ja span -elementtien sisällä.

Sisennysten edut tulevat erityisen hyvin ilmi, kun johonkin elementtiin joutuu jatkuvasti viittaamaan. Näin esimerkiksi eri sivupohjat määrittelevät luokat voidaan pitää erillään toisistaan ja niihin vaikuttavia tyylimuutoksia voidaan kirjoittaa yksien kaarisulkeiden sisään. [10.]

Muuttujilla voidaan määritellä esimerkiksi ulkoasun tärkeimmät värit, joita joudutaan käyttämään jatkuvasti. Sekoituksilla voidaan käyttää jo aiemmin määriteltyjä tyylejä uudestaan johonkin toiseen elementtiin. [10.]

Kuvat ja muut mediat

Kuvien osalta responsiivisuus saadaan toteutettua hyvin yksinkertaisesti, kuten esimerkissä 5:

```
img{
max-width:100%;
}
```

Koodiesimerkki 5. Yleinen CSS -tyyli responsiivisille kuville.

Esimerkin 5 kaltainen CSS-määrittely kuville antaa niille maksimiarvoksi 100 %, mutta jos tila on pienempi, ne mukautuvat pienemmäksi säilyttäen kuitenkin mittasuhteet [6, s. 45].

Kuvien osalta saattaa olla myös järkevää jakaa erikokoisia kuvia eri laitteille. Mobiililaitteet pärjäävät hyvin hieman pienemmillä kuvilla. Kuvien skaalaus CSS:n avulla ei vaikuta ladattavan kuvan kokoon, ja näin ollen kuva, joka on tarkoitettu katseltavaksi koko leveydessä isolla näytöllä, latautuu myös pienille mobiilinäytöille. Tämä lisää datan käyttöä turhaan heikommilla mobiili-internetyhteyksillä. Tälle tekniikalle on keksitty englannin kielinen termi ”adaptive images” tai mukautuvat kuvat. Mukautuvat kuvat voidaan toteuttaa monella eri tapaa, joko selainpuolella tai palvelinpuolella. [11.]

Muiden medioiden osalta responsiivisuus ei toteudu aivan niin yksinkertaisesti. Esimerkiksi yleisesti käytetyt videoiden upotukset sivustoilla eivät mukaudu pienille näytöille yhtä yksinkertaisesti kuin kuvat. Videoiden upotus esimerkiksi Youtube-palvelusta tehdään iframe-tekniikkaa käyttäen, jonka tarkoituksena on hakea sisältö eri sivustolta. Kun video tulee kolmannelta osapuolelta, sen tyyleihin tai muuhun esitykseen voi vaikuttaa vain palveluntarjoajan tarjoaman rajapinnan avulla. CSS:ää käyttävä tekniikka on olemassa, mutta se vaatii sitä käyttävien videoiden olevan samaa kuvasuhdetta. Videot tai mikä tahansa muu sisältö on iframe-elementillä toteutettuna tarvitsee aina korkeuden määreen tai muuten se palautuu oletusasetuksena 150 pikselin korkeuteen. Tämä aiheuttaa videoiden kohdalla ongelman, jossa selain ei

ymmärrä pitää kuvasuhdetta paikoillaan, koska korkeus on määritelty. Videon sisältävä iframe-elementti voidaan ympäröidä div-elementille, jolla määritellään padding-bottom-arvo, joka on kuvasuhteen suhdeluvun mukainen prosenttimäärä, eli esimerkiksi 16/9-kuvasuhteella 56.25%. Lisäksi diville annetaan ”position: relative;” -määritys ja divin sisältämälle iframe-elementille annetaan määreet ”position: absolute; top: 0; left: 0;”. Näin ollen ulompi div-elementti toimii tavallaan oikean muotoisena ikkunana iframe-elementille, joka peittää kaiken tarjolla olevan tilan. [12.]

Jos video on omalla palvelimella eikä kolmannen osapuolen palvelussa, videoiden esittäminen responsiivisesti internetsivulla on lähes yhtä mutkatonta kuin kuvien esittäminen. Tällöin voidaan käyttää HTML5:n tarjoamaa video-elementtiä, jonka sisään määritellään videon lähde source-elementtiin. Ainoastaan videon kokoon on syytä kiinnittää huomiota, sillä mobiilikäyttäjät eivät tarvitse yhtä korkealaatuista resoluutiota kuin esimerkiksi työpöytätietokoneiden käyttäjät. Mobiilikäyttäjien internetyhteydet ovat usein myös rajoitettuja joko nopeuden tai määrän kannalta, jolloin turhan isosta videotiedostosta on vain haittaa.

Iframeja ei käytetä pelkästään videoiden upottamiseen, vaan niillä voi upottaa sivulle mitä tahansa sisältöä, kuten esimerkiksi Facebook-tykkäyksistä kertovan elementin. Ongelmalliseksi responsiivisuuden kannalta iframet tekee juuri niissä olevan upotettavan materiaalin staattinen koko.

JavaScript ja responsiivisuus

Pelkän HTML:n ja CSS:n avulla pystyy toteuttamaan suurimman osan responsiivisista ominaisuuksista, mutta ajoittain tulee vastaan tapauksia, joihin nämä tekniikat ei riitä. Tyypillisesti JavaScriptiä tarvitaan tiettyihin navigaatioratkaisuihin. Vaikka CSS:n avulla voidaan näyttää ja piilottaa elementtejä, jotkin elementit tarvitsevat silti toimintoja jotka toteutetaan JavaScriptin avulla. Lisäksi JavaScriptillä saadaan myös tieto käyttäjän ikkunan koosta, jos CSS:n mediakyselyt eivät riitä. Tällaisia tapauksia ovat esimerkiksi elementit, joiden mittoja ei jostain syystä voi määritellä pelkän CSS:n avulla.

Sivustolle saatetaan haluta toteuttaa erilainen JavaScript toiminnallisuus riippuen siitä, minkäkokoista laitetta käyttäjä käyttää. JavaScriptillä voi ikkunan koon saada selville helposti, mutta jos näyttöpinta-ala muuttuu siihen pitää pystyä reagoimaan. Näytön kokoon reagoiminen vie luonnollisesti enemmän tietokoneen resursseja kuin

mediakyselyt, joten niiden käyttämisessä on oltava tarkkana. JavaScriptillä voi hakea tiedon käyttäjän ikkunan koosta seuraavasti: ”document.documentElement.clientWidth”. Tätä menetelmään voidaan käyttää hyödyksi esimerkiksi tarkastelemalla käyttäjän ikkunan kokoa sivun latautuessa. [13.]

JavaScriptiä tarvitaan myös joihinkin mobiilivalikoiden toteutuksiin. Pienellä näyttöpinta-alalla halutaan säästää tilaa, ja valikko on yksi elementti, joka olisi hyvä olla helposti saatavilla, mutta toisaalta se vie tilaa sisällöltä. JavaScriptin avulla toimivissa valikoissa itse valikko voidaan näyttää tai piilottaa jotakin elementtiä painamalla. Navigaation piilotus ja näyttäminen on yksinkertaista toteuttaa esimerkiksi jQuery-kirjastolla (http://api.jquery.com/).

Responsiivisia sivuja tehtäessä on otettava huomioon, että mobiililaitteiden JavaScript-tuki ei ole samalla tasolla tehokkaampien työpöytäkoneiden kanssa. Suorituskykyongelmiin voi törmätä esimerkiksi intensiivistä ja toistuvaa laskentaa vaativissa JavaScript toteutuksissa.

4.2 Responsiivisen internetsivun suunnittelu

Responsiivisen internetsivun suunnittelussa on monia lähestymistapoja, joista muutamat ovat kasvattaneet suosiotaan muita enemmän. Esimerkiksi ”mobile first” eli mobiili ensin -suunnittelustrategia pyrkii luomaan kaikilla laitteilla toimivan perussivun, jota voidaan laitteiden ominaisuuksien puitteissa parantaa (*engl. progressive enhancement*). Mobile first toteutetaan nimensä mukaisesti mobiilista lähtien, eli tyylit, jotka eivät ole mediakyselyiden piirissä olettavat, että käyttäjällä on vähän näytön tilaa käytössä. Mediakyselyt rajataankin niihin kohtiin, jossa uutta näyttöpinta-alaa voidaan järkevästi hyödyntää. Mobiilille ensiksi suunnitellun sivun vahvuuksista yksi on, että se toimii jopa kaikkein erikoisimmissa laite- ja selain- yhdistelmissä. Esimerkiksi JavaScriptin tai uusimpien CSS-ominaisuuksien tuki ei estä sivuston käyttöä. [1.]

Perinteisiin, yhdelle koolle tehtyihin sivuihin verrattuna responsiivinen sivu vaatii erilaiset lähtökohdat. Ennen graafikko on saattanut piirtää yhden luonnoksen sivusta ja ohjelmoija on toteuttanut sen pikselin tarkkuudella. Responsiivisissa sivuissa on kuitenkin suuri määrä mahdollisia näkymiä, ja kaikkia niitä ei ole järkevää piirtää erikseen. Usein lähtökohtana on kolme eri näkymää: työpöytä, tabletti ja mobiili.

Näiden välimaastoon sijoittuvat laitteet voidaan pitää toimivina käyttäen mittasuhteita tarkkojen pikselimäärien sijaan. Ethan Marcotte suosittelee lähes kaiken sisällön tekemistä mittasuhteiden avulla. Mittasuhteita käytettäessä tekstin asemointiin ja palstan leveyteen on syytä kiinnittää erityistä huomiota. Mittasuhteita käytettäessä palstan leveys saattaa leveillä näytöillä venyä liian pitkäksi ja näin ollen vaikealukuiseksi, ellei mittoja rajoiteta max-width-asetuksella. [1; 3.]

Responsiivisten projektien hallintaan soveltuvat erinomaisesti ketterät menetelmät. Projekteille on yleensä tyypillistä, että niiden vaatimukset elävät vielä toteutuksen aikana ja kaikkia mahdollisia tapauksia on vaikea ottaa huomioon. SCRUM on sovelluskehitysmalli, joka näkee ohjelmointiprojektin koostuvan sykleistä. Lisäksi joka päivä pidetään enintään 15 minuutin pituinen palaveri, jossa jokainen projektin toteutukseen osallistuva jäsen kertoo, mitä on tehnyt edellisenä päivänä, mitä aikoo tehdä seuraavan päivän aikana ja mitä mahdollisia ongelmia tai esteitä on noussut esiin tavoitteen saavuttamisessa. SCRUMissa projektimäärittely kehittyy tavallisesti projektin edetessä. [14.]

Responsiivisen toteutuksen alkaessa jatkuva ja joustava projektinhallinta on ensiarvoisen tärkeää. SCRUMin päivittäiset palaverit ovat oiva tapa työstää käyttöliittymää, jonka pitää olla mahdollisimman mukautuva. Ongelmakohdat joissakin tietyissä näytön ko'oissa saadaan ratkottua nopeasti, eikä yhden toiminnallisuuden kehitys hidastu turhaan. Responsiivisissa projekteissa tulisi myös olla määriteltynä, missä kohtaa projekti on valmis. Esimerkkinä ”valmiin” projektin määritelmästä voisi olla esimerkiksi lista selaimista ja laitteista, joilla lopputulos toimii.

Responsiiviset sommittelumallit

Responsiivisessa suunnittelussa on muutamia hyväksi todettuja suunnittelumalleja. Tavalliset elementit, kuten otsikko, leipäteksti ja tekstin seassa olevat kuvat, on helppo latoa allekkain responsiivisesti, mutta monimutkaisemmat elementit, kuten navigaatio, ovat hiukan haastavampia.

Haasteellisten elementtien toteutukseen on muutamia vakiintuneita käytäntöjä. Navigaation voi toteuttaa monella tapaa, riippuen navigaatiossa olevista tasoista ja sivujen määrästä, mutta suurimmalla osalla malleista on yhteistä se, että ne vievät mahdollisimman vähän tilaa silloin, kun navigaatiota ei tarvita. Käyttäjän kannalta

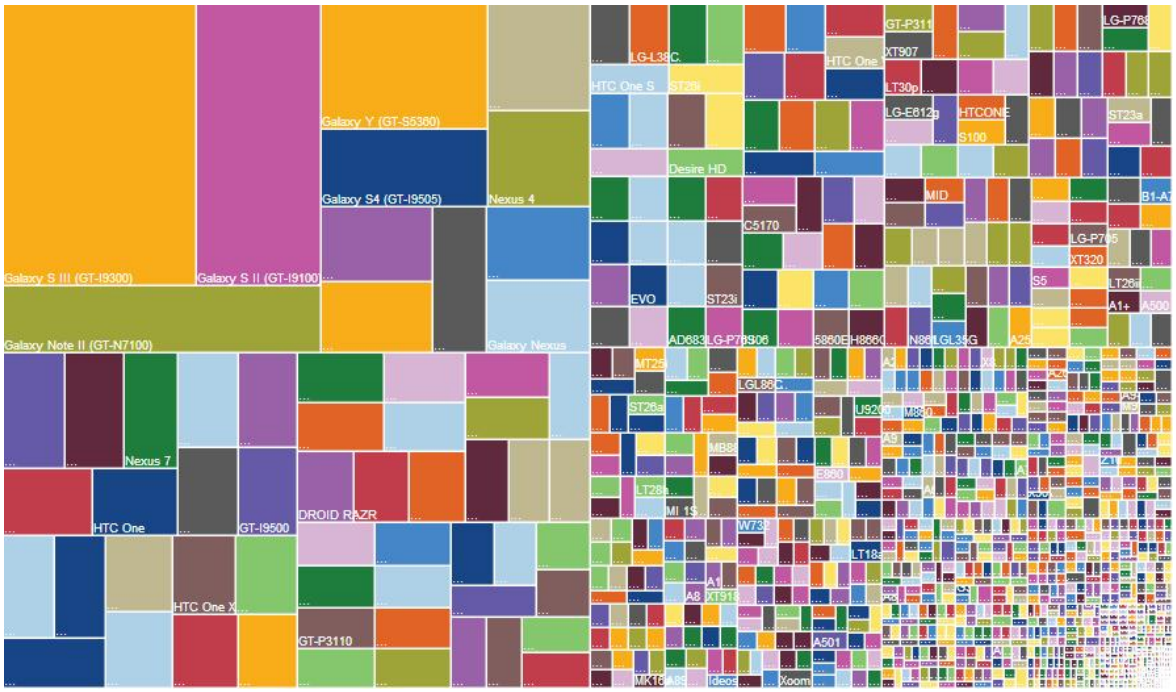
oleellinen sisältö halutaan tuoda mahdollisimman helposti luettavaan muotoon, ja siksi paljon tilaa vievät navigaatioratkaisut eivät sovi mobiililaitteisiin. [14.]

Sommittelussa yksittäiset kuvat on helppo saada mukautumaan, mutta se ei välttämättä riitä, jos kuvia tahdotaan monta vierekkäin. Lisäksi ongelmia saattavat aiheuttaa kuvien eri mittasuhteet. Mobiilinäkymässä ryhmän kuvia saattaakin joutua pilkkomaan pienempiin ryhmiin, esimerkiksi neljän kuvan ryhmän pienemmiksi kahden kuvan ryhmiksi allekkain. Tämä onnistuu yksinkertaisimmillaan CSS:n float-määrittelyllä ja :nth-child-pseudoluokkaa hyväksi käyttämällä. Pienille laitteille voidaan tarjota CSS-tyyli, joka antaa esimerkiksi joka toiselle kuvalle halutun CSS-tyylin, käyttäen hyväksi img:nth-child(odd)-valintaa, kun taas oletustyyli muille laitteille voitaisiin asettaa valinnalla img:nth-child(4n), joka siis valitsee vain joka neljännen elementin. [8.]

5 Responsiivisten sivustojen testaus

Internetsivujen selaintestaus oli ennen responsiivisia sivuja melko yksinkertaista. Ennen mobiililaitteita internetsivuja piti testata lähinnä suosituimmilla selaimilla ja yhdellä koolla. Selaimet, joilla internetsivujen piti toimia saattoivat rajoittua Firefox-, Safari- ja IE6 -selaimiin. Vanhat Internet Explorer selaimet, kuten IE6, olivat toki hankalia puutteellisten standardien tuen takia, mutta testaus oli silti melko yksinkertaista.

Responsiivisten sivujen ja mobiililaitteiden myötä selaintestaus on muuttunut. Varmasti yhdelläkään kehittäjällä ei ole käytettävissään kaikki mahdollisia laitteita ja mobiiliselaimia. Esimerkiksi pelkästään Android-laitteiden pirstaloitumista havannoillistaa kuva 3.

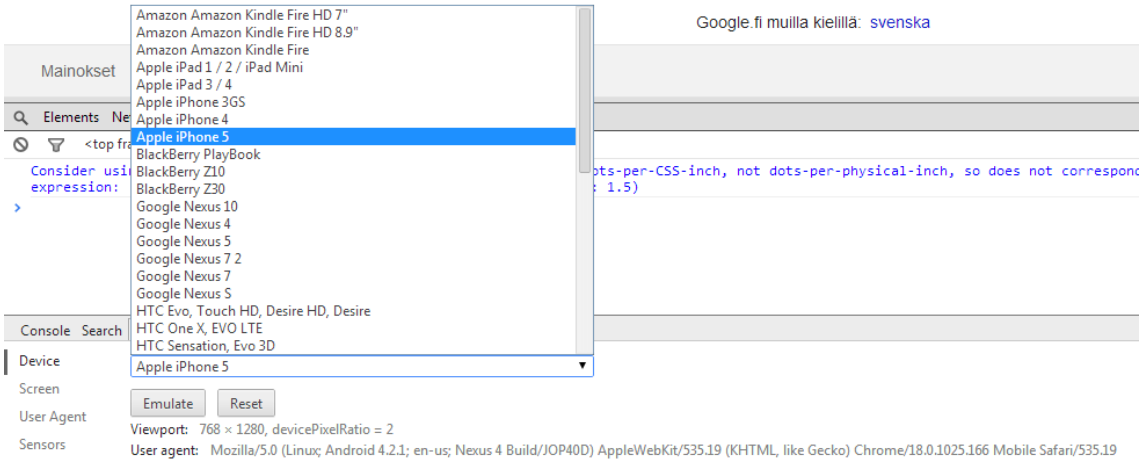


Kuva 3. Android-alustan pirstaloituminen. Jokainen laatikko esittää markkinoilla olevaa Android-laitetta ja niiden markkinaosuutta[15].

Kaikkien erilaisten laitteiden määrästä voi jo päätellä, että responsiivisten sivustojen kattava testaus on hankalaa. Esimerkiksi Windows Phone 7.8 -version Internet Explorer -selain ei tue muita kuin oletusfontteja. Koska kaikkia laitteita on

epäkäytännöllistä testata, tulee toteutuksen laatuun panostaa. W3C-standardin mukainen HTML-rakenne, laadukas JavaScript-koodi sekä CSS-tyylit, joissa on otettu huomioon vanhempien selainten tuki, voivat vähentää virheiden määrää. [15.]

Responsiivisen sivuston käyttäytymisen saa helposti selville muuttamalla selainikkunan kokoa, mutta laitekohtaiset erot eivät käy tästä ilmi. Chrome-selaimen kehittäjätyökalujen ollessa päällä selainikkunan oikeaan ylälaitaan ilmestyy ikkunan koko pikseleinä, kun ikkunan kokoa muuttaa. Kuten kuvasta 4 näkyy, Chromessa on kattava valikoima laiteprofiileja, joiden avulla kokoa, selaintunnisteita ja kosketustapahtumia voi emuloida. [16.]



Kuva 4. Chromessa valmiina olevia laiteprofiileja

Myös uudemmissa Internet Explorer-selaimissa on mahdollisuus emuloida Microsoftin omia käyttöjärjestelmiä. Emuloinnilla voi päästä melko lähelle autenttisia laitteita, mutta ne eivät voi niitä kuitenkaan korvata. Esimerkiksi Internet Explorerista löytyvä vanhempien IE-versioiden emulaattori ei ymmärrä ehdollisia CSS-viittauksia, jotka ovat olennainen osa vanhempien IE-selainten toimintaa responsiivisilla sivuilla.

Riippuen kehitysympäristöstä sivun testaus autenttisella laitteella voi olla hankala prosessi. Kehittäjän paikallisessa ympäristössä muutokset saadaan nopeasti näkyviin, mutta kun halutaan saada sivu näkymään oikealla laitteella, täytyy muutokset aina viedä palvelimelle. Lisäksi pienemmissä projekteissa ei välttämättä ole järkevää järjestää palvelinta pelkästään testausta varten, jolloin pahimmassa tapauksessa oikeilla laitteilla testaus joudutaan suorittamaan vasta, kun sivusto on julkinen. Tämän

insinöörityön projekteissa sisäinen testaus toteutettiin palvelimella, johon oli pääsy vain konsernin Active Directory -tunnuksilla. Ennen tuotantopalvelimelle vientiä Creunan testauksessa kuuluu myös niin sanotulle staging site -palvelimelle vienti, joka on samankaltainen tuotantoympäristön kanssa. Staging-vaiheessa sivulle tehtiin muun muassa JavaScript- ja CSS-tiedostojen pakkaus. [16.]

6 Kolmen reponsiivisen sivuston suunnittelu ja toteutus

Tämä osa insinöörityöstä on salainen.

7 Johtopäätökset ja yhteenveto

Responsiivisen verkkosivuston toteutus eroaa perinteisen verkkosivuston toteutuksesta paljon. Responsiivisten internetsivujen mukautuvuuden takia joudutaan entistä useampaan asiaan ottamaan kantaa jo suunnitteluvaiheessa. Suunnittelussa responsiivisessa kehityksessä on otettava huomioon lopputuloksen vaihtelevat mitat, ja suunnittelun on myös oltava joustavaa. Perinteiset työskentelytavat eivät aina sovi responsiivisiin toteutuksiin, esimerkiksi ulkoasun suunnittelun osalta, vaan uudet tavat, kuten nopean prototyypin kehitys, saattavat olla perusteltuja. Responsiivisiin toteutuksiin soveltuvat strategiat, kuten mobile first, tukevat responsiivista kehitystyötä ja johtavat usein parempaan lopputulokseen. Toteutuksessa voivat auttaa myös ketterät menetelmät, esimerkiksi scrum. Scrumin päivittäiset palaverit tukevat responsiivisessa kehityksessä nopeasti eteen tulevien tilanteiden ratkaisua.

Reponsiivisen toteutuksen mahdollistavia tekniikoita ovat tärkeimpänä CSS:n mediakyselyt, joilla voidaan antaa erikokoisille päätelaitteille eri tyylejä. Responsiivisessa toteutuksessa pitää kuitenkin ottaa huomioon myös HTML-rakenne ja JavaScript. Valmiit CSS-kirjastot tai SASS voivat auttaa kehittäjää toteuttamaan responsiivisuutta nopeammin.

Responsiivisten sivujen testaus on muuttunut monimutkaisemmaksi. Laitteiden suuren määrän vuoksi testauksessa ei voida aina olla täysin varmoja, miltä sivusto tulee näyttämään oikealla, fyysisellä laitteella. Erilaiset emulaattorit ja valmiit laiteprofiilit, joita on esimerkiksi Chrome- ja IE-selaimissa, auttavat responsiivisten sivujen testauksessa.

Insinöörityössä tehtyjen projektien toteutukset onnistuivat vaihtelevasti. Yritys A:n ensimmäisen tuotteen kampanjasivussa oli selvästi käyttöliittymän kehityksen osalta parannettavaa työtavoissa. Pieneen kampanjasivuprojektiin on järkevää käyttää valmista CSS-kehystä, kuten Twitter Boostrapia (http://getbootstrap.com/). Projektin kesto venyi suunnitellusta noin kaksinkertaiseksi. Staattisen prototyypin kehitys ilman responsiivista CSS-kehystä onnistui hyvin, mutta kun prototyyppi siirrettiin osaksi sisällönhallintajärjestelmää ja sivun testaus käynnistyi monilla eri laitteilla, olisi osa ilmi tulleista virheistä voitu ohittaa valmiilla CSS-kehyksellä. Projektin kilpailuun tulleet osallistumiset olivat laadultaan erinomaisia, ja asiakas pystyikin käyttämään osallistumisia hyväkseen sosiaalisessa mediassa markkinointiin. Vaikka projektin toteutus viivästyi, se valmistui kuitenkin ennen tuotteen julkaisemista.

Yritys A:n jälkimmäisen kampanjasivun toteutuksessa otettiin opiksi ensimmäisen kampanjasivun toteutuksesta ja käytettiin CSS-kehystä. Twitter Bootstrap (http://getbootstrap.com/) CSS-kehys teki prototyypin ja valmiin sivun kehityksestä nopeaa. CSS-kehyksen käyttäminen ennaltaehkäisi sivun skaalautumisessa ilmenneitä ongelmia, joita ensimmäisen tuotteen kampanjasivun projektissa esiintyi. Toisen tuotteen kampanjasivu oli hyvin samankaltainen ensimmäisen kanssa, mutta sen toteutus sujui huomattavasti nopeammin.

Yritys B:n projektin osalta toteutus oli haasteellinen. Aikataulu oli tiukka, mutta se piti suurimmaksi osaksi. Tiukan aikataulun vuoksi projektia ei ehditty suunnitella niin tarkkaan kuin olisi ehkä ollut tarpeellista. SASSin käyttäminen helpotti CSS-tyylien kirjoittamista, mutta toisin kuin responsiiviset CSS-kehykset, se ei tarjoa valmiita luokkia responsiivisuuden toteuttamiseen. Verkkokauppa ja sen toteutus onnistui käyttöliittymän osalta hyvin, ja koko maksuprosessi oli mahdollista suorittaa myös mobiililaitteilla. Myös muut sivupohjat toimivat mobiililaitteilla. Sivustolle toteutettu mobiilinavigaatio osoittautui toimivaksi ratkaisuksi. Projekti oli tärkeä asiakkaalle, sillä siinä toteutettiin liiketoiminnalle kriittinen verkkokauppa. Projektin aikataulussa tärkeimmät päämäärät saavutettiin ajoissa.

Responsiivinen suunnittelu on nopeasti muodostumassa internetsivustojen pääasialliseksi toteutustavaksi. Erikokoisten näyttöjen laaja kirjo, mobiiliyhteyksien nopeuksien kasvu ja kuluttajien odotukset käyttökokemuksen osalta pakottavat internetsivujen kehittäjät luomaan palveluja, jotka ovat mahdollisimman mukautuvia. Kehittäjät joutuvat vielä toistaiseksi paikkaamaan vanhentuneiden selainten tukea uusille teknologioille erilaisten JavaScript-kirjastojen avulla, mutta selainten ja laitteiden muuttuessa koko ajan modernimpaan suuntaan, voidaan tulevaisuudessa hyödyntää entistä enemmän uusien tekniikoiden mahdollistamia ominaisuuksia.

Lähteet

1 Wroblewski, Luke. 2011. Mobile first. New York: A Book Apart.

2 Datta, A. 2012. Be Responsive: A History of Responsive Design. Verkkodokumentti. < http://shout.setfive.com/2012/03/12/be-responsive-a-history-of-responsive-design/>. Luettu 11.9.2014.

3 Media Queries. 2012. Verkkodokumentti. World Wide Web Consortium <http://www.w3.org/TR/CSS3-mediaqueries/>. Luettu 11.9.2014

4 Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2013–2018. 2014. Verkkodokumentti. Cisco. <http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white_paper_c11-520862.HTML>. Luettu 31.3.2014.

5 Marketing in a multiscreen world. 2014. Verkkodokumentti. Millward Brown. <http://www.millwardbrown.com/adreaction/2014/#/main-content>. Luettu 31.3.2014.

6 Marcotte, Ethan. 2011. Responsive web design. New York: A Book Apart.

7 HTML5: A vocabulary and associated APIs for HTML and XHTML. 2014. Verkkodokumentti. World Wide Web Consortium. <http://www.w3.org/TR/HTML5/introduction.HTML>. Luettu 3.4.2014.

8 Cascading Style Sheets (CSS) Snapshot 2010. 2011. Verkkodokumentti. World Wide Web Consortium. <http://www.w3.org/TR/CSS-2010/>. Luettu 3.4.2014.

9 Otto, M., Thornton, J. Bootstrap. Verkkosivu. <http://getbootstrap.com/2.3.2/>. Luettu 1.4.2014.

10 Weizenbaum, N. 2014 SASS (Syntactically Awesome StyleSheets). Verkkodokumentti. <http://SASS-lang.com/documentation/file.SASS_REFERENCE.HTML>. Luettu 3.4.2014.

11 Responsive Images: Experimenting with Context-Aware Image Sizing. 2010. Verkkodokumentti. Filament group. <http://filamentgroup.com/lab/responsive_images_experimenting_with_context_aware_image_sizing/>. Luettu 3.4.2014.

12 Coyier C. 2012. Rundown of Handling Flexible Media. Verkkodokumentti. <http://CSS-tricks.com/rundown-of-handling-flexible-media/>. Luettu 16.9.2014

13 Web API interfaces. 2014. Verkkodokumentti. Mozilla developer network. < <https://developer.mozilla.org/en-US/docs/Web/API>. Luettu 3.4.2014.

14 Ketteryys haltuun: Scrum pähkinänkuoressa. Verkkodokumentti. Sininen Meteoriitti. <http://www.meteoriitti.com/Artikkelisarjat/Ketteryys-haltuun/Ketteryys-haltuun-Scrum-pahkinankuoressa/>. Luettu 31.3.2014.

15 Android Fragmentation Visualized. 2013. Verkkodokumentti. OpenSignal. <http://opensignal.com/reports/fragmentation-2013/>. Luettu 31.3.2014.

16 Mobile emulation. 2014. Verkkodokumentti. Google. <https://developers.google.com/chrome-developer-tools/docs/mobile-emulation?hl=fi>. Luettu 31.3.2014

17 Visual Studio. 2014. Verkkosivusto. Microsoft. <http://www.visualstudio.com/>. Luettu 3.4.2014.

18 Weaver, J. Off canvas. Verkkodokumentti. <http://jasonweaver.name/lab/offcanvas/>. Luettu 3.4.2014.